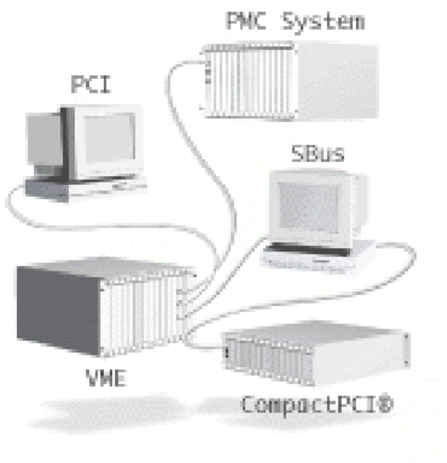
SedNet™ 200 Mbps IEEE-1394 VMEbus Master Adapter

with optional IP module carrier board

SD-VME-200-2

User's Manual v1.1



HOW TO REACH US?

Sederta Inc.
5065 Levy, St-Laurent,
Québec, Canada, H4R 2N9

For information, comments or suggestions, contact us by E-mail at info@sederta.com

For technical support, contact us by:

Tel.: (514) 338-8749
Fax: (514) 338-1915
E-mail: support@sederta.com

Visit our web site for more information about our company and SedNet products at:

http://www.sederta.com

Disclaimer

Table of contents

List of tables

List of figures

1. INTRODUCTION

1.1 SedNet Adapter Card Features

<u>1394 section</u>

- 100 and 200 Mbps data communication transfer.
- Physical isolated interface.
- Asynchronous and isochronous communication.
- 3 IEEE-1394 connectors.
- 1394 hot plug & play.
- Compatible with SedNet™ adapters & libraries.
- Up to 64 nodes connected.

<u>User memory section</u>

- 256KBytes of 32 bit Fast SRAM, 0-wait state.
- 256KBytes of 8 bit Flash memory.
- 32MBytes of DRAM.

<u>Firmware section</u>

- Complete resident firmware, intelligent bridge between 1394, VME and local buses.
- Real-time specific functions :
  Analog and Digital provider,
  Programmable isochronous timer to synchronize all attached nodes and IP modules,
  Specific Real-time functions.
- Capabilities to add specific functions.

<u>VMEbus section</u>

- Complete VMEbus Interface Controller and Arbiter.
- Complete VMEbus 32 bits Master and slave Interface.

<u>IP Carrier section</u>

- Up to 12 Industrial Pack sites (32MHz or 8MHz) using 6U IP Module Carrier Boards.
- Interrupt and DMA transfer supported.
- Real-time synchronization between all attached devices and all IP modules.

<u>Hardware Specifications</u>

- Length: Standard VME 6U
- Storage Temperature: 0°C to 70°C
- Operating Temperature: 5°C to 55°C
- Operating Humidity: 20% to 90% non-condensing
- Operating Voltage: +5,+/-12V, +/-5%

1.2 Basic Functionality overview

1.2.1 Block diagram of the 1394-VME master adapter

The SedNet 1394-VME master adapter is a high performance intelligent bridge between the 200 Mbps 1394 high speed serial bus, the VME bus and the Industrial Pack (I.P.) Modules. It also contains a large amount of user resident memory (FLASH, SRAM, DRAM) and has a RS232 port for debugging and configuration.

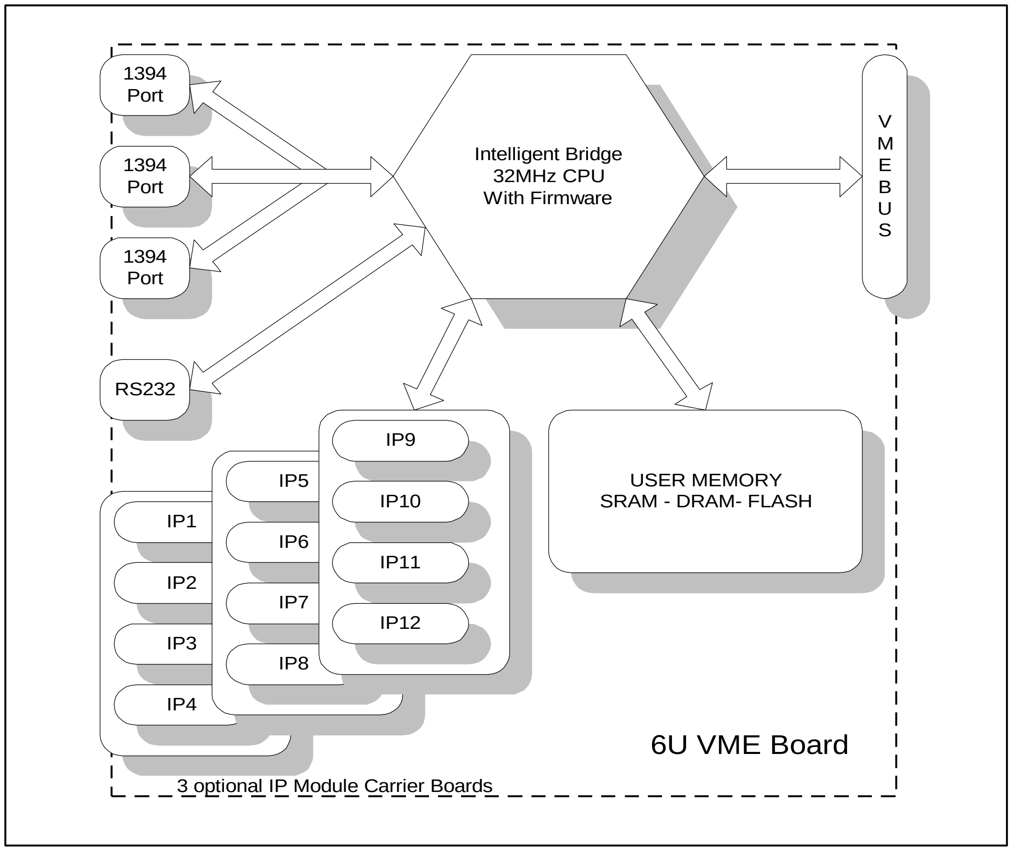


Figure 1: Block diagram of the 1394-VME master adapter

The SD-VME-200-2 includes a 32-bit integrated processor unit, the MC68360 running at 32 MHz. The processor combines high-performance data manipulation capabilities with powerful on-chip peripheral subsystems. The board includes 256KB of user EPROM FLASH memory, 256KB of user SRAM (static RAM), and 32MB of SIMM DRAM. The card interfaces up to twelve IP modules with the IP carrier board.

The RS-232 port is mainly used for system configuration and system debugging. It can be plugged to any serial communication device such as a modem, a printer or a terminal in order to exchange or display data. It can also be used to monitor the system activity and as an easy way to reconfigure your system. The configuration of the SD-VME-200-2 board can also be done by the 1394 bus or by the VME bus.

1.2.2 Accessing the 1394-VME master adapter from the 1394 bus

The SD-VME-200-2 board maps on the 1394 bus all the VME memory space and its local memory including the user memory and the IP module.

The SD-VME-200-2 board can be accessed by the other 1394 devices in order to perform read or write operations to the local memory of the 1394-to-VME bridge or to the VME bus.

Sending a 1394 request to the 1394-VME bridge performs all these different accesses. The main difference between the two different accessed zone is the address offset high (16 bits) in the header of the 1394 packet, as described in the table below.

Address High	**Destination Address**
0x0000	Local memory space
0x001A	VME bus address space

Table 1: Destination address depending on the packet's address offset high

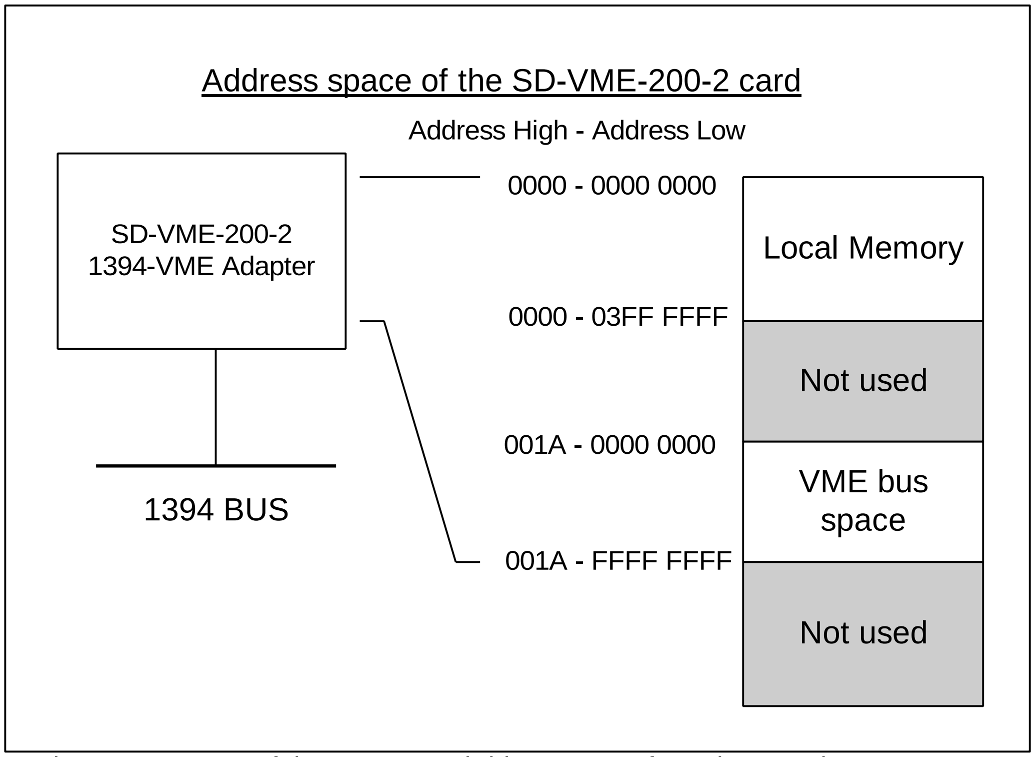


Figure 2: Memory of the VME-1394 bridge as seen from the 1394 bus

1.2.3 Accessing the 1394-VME master adapter from the VME bus

The boards connected on the same VME bus can make two different kinds of access to the 1394-VME master adapter:

- Direct access to the local memory mapping using the VME address masks.
- Access to the 1394 bus (by sending a request read or write packet to the 1394-VME master adapter). Sending pre-formatted 1394 request packets to the 1394-bridge performs these accesses. (see section 3 and Figure 3-1 for more details on 1394 Request packets by the VME bus)

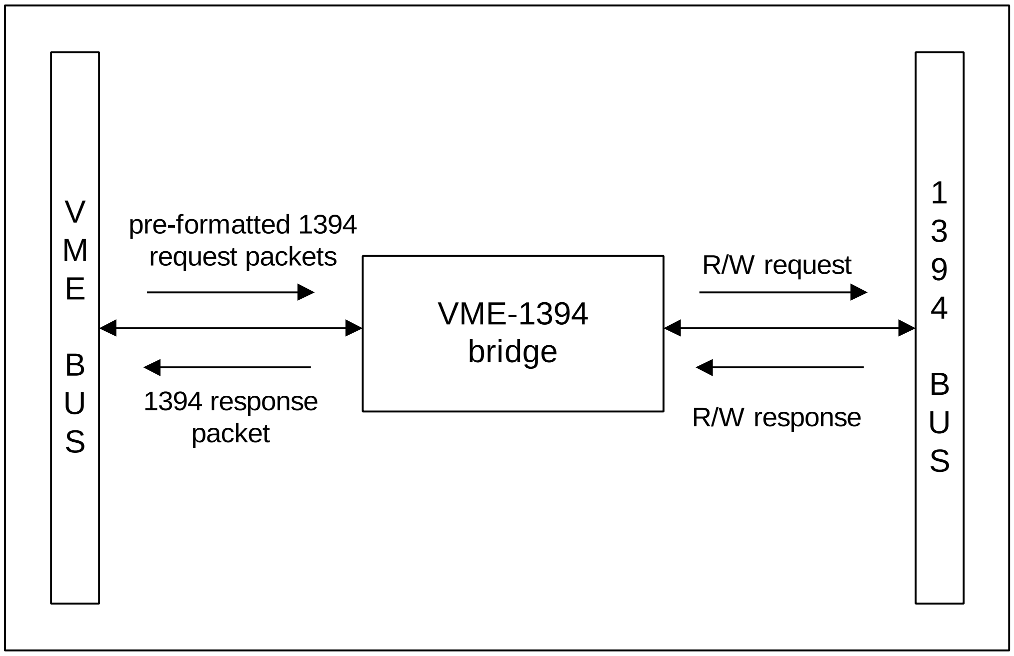


Figure 3: 1394 access from the VME bus

1.3 Architecture Overview

The SD-VME-200-2 block diagram is shown in Figure 1-2.

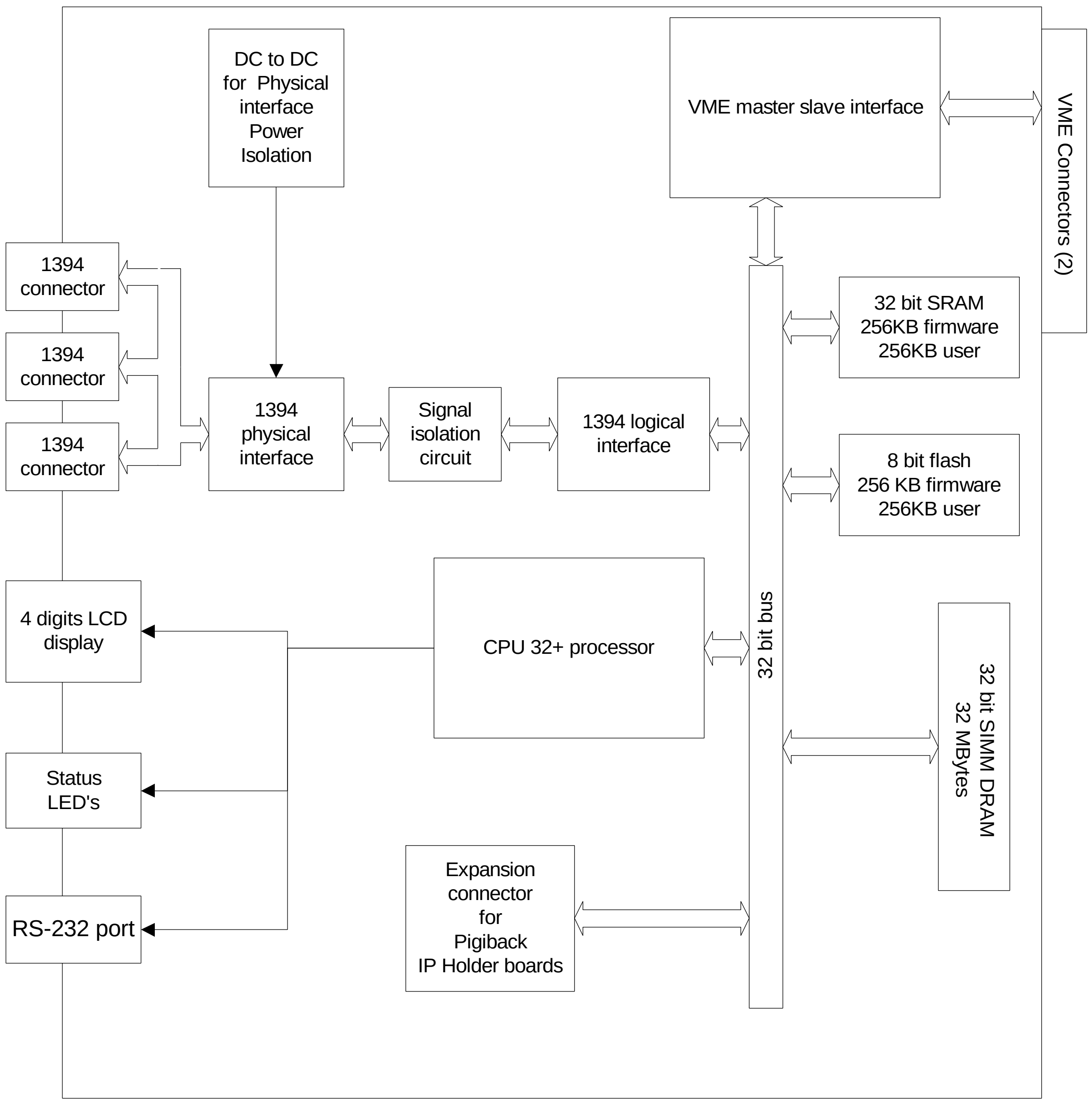

DC to DC for Physical interface Power Isolation
VME master slave interface
VME Connectors (2)
1394 connector
1394 connector
1394 connector
1394 physical interface
Signal isolation circuit
1394 logical interface
32 bit SRAM 256KB firmware 256KB user
8 bit flash 256 KB firmware 256KB user
32 bit bus
4 digits LCD display
CPU 32+ processor
32 bit SIMM DRAM 32 MBytes
Status LED's
RS-232 port
Expansion connector for Pigiback IP Holder boards

Figure 4: Functional diagram of the SedNet VME adapter card

1.3.1 MC68EN360 CPU

The MC68360 controller is a Motorola CPU32+ family member. The CPU 32 main processor has a 32-bit data path and can address up to 4 GBytes of memory space. There's also an on-chip RISC processor that acts as a dedicated communications coprocessor, for things such as DMA, serial communications, timers and more.

1.3.2 1394 serial bus interface

The SD-VME-200-2 uses the TSB12LV01 as a link layer to transmit and receive 1394 packets. It also generates and checks the 32-bit CRC. The link layer only provides a half-duplex (transmit or receive) data packet delivery service. The process of delivering a single packet is called a «subaction». Two types of «subaction» are used in the Serial Bus link layer.

a) An asynchronous subaction -- a variable amount of data and several bytes of transaction layer information are transferred to an explicit address and an acknowledge is returned

b) An isochronous subaction -- a variable amount of data is transferred on regular intervals. No acknowledge is returned.

The 1394 link layer has software-adjustable FIFO for packet size and performance characterization. It interfaces directly to its companion chip, the TSB21LV03A physical layer. The TSB21LV03A can support bus speeds of 100 & 200 Mbytes/sec.

The physical layer provides the analog transceiver functions needed to implement a three-port node in a cable-base 1394 bus. The three fully IEEE-1394 compliant cable ports provide a bandwidth of 200 Mbits/sec. Its logic performs system initialization and arbitration functions.

1.3.3 VME interface

The VME interface is composed of the VME master/slave interface chip (Cypress VIC068) and of a programmable logic chip.

The VME interface allows the card to become VME bus controller in order to perform bus arbitration like slave selection, bus grant, acknowledge, etc…

The programmable logic chip is used for address decoding and mapping of the adapter card on the VME bus.

1.3.4 Memory block

The SD-VME-200-2 card is delivered with the following memory units:

- 256 Kbytes of 8 bits program FLASH
- 256 Kbytes of 8 bit user FLASH
- 256 Kbytes of 32 bit program SRAM
- 256 Kbytes of 32 bit user SRAM
- 32 Mbytes of 32 bit (SIMM) user DRAM

1.3.5 4 digits LCD display

The LCD display is to indicate the state of the system. Messages will be displayed when hardware tests are performed, or when a system fault occurs. The LCD display is also used to display the 1394 node number of the adapter card.

1.3.6 Status LED

Four LEDs are used on the card’s front panel:

- the Run LED
- the Idle LED
- the Error LED
- the SCon LED

The Run LED is lit when the 1394 interface is active. It indicates that a 1394 packet is being sent or received at that moment.

The Idle LED is illuminated when no access to the 1394 or to the VME bus is performed.

The Error LED is flashing when an unrecoverable error has occurred the processor will then be in some HALT state. When a minor error has occurred the LED will be ON until the cause of the error has been deactivated. At system RESET, the Error LED should be OFF if the card is running under normal conditions.

The SCon LED is the VME system controller indicator. When the LED is ON, the adapter card is supposed to be in slot 1 of the VME bus and has the capability of being bus controller.

1.3.7 RS-232 port

The RS-232 port is used as a debugging tool for system functionality such as 1394 quadlet write and read. It can be used as a normal RS-232 port for a modem or a serial printer and can also be used to configure the system’s parameters such as 1394 node number and VME mapping of the card. Finally, a terminal screen can be connected to the port so it becomes a monitor for system activity.

1.3 1394 overview

1.3.1 Introduction to 1394

There are two kinds of packet delivery (called ”subaction”) used by the 1394 Serial Bus link layer:

-Asynchronous subaction

-Isochronous subaction

1.3.1.1 Asynchronous Subaction

Asynchronous arbitration is used whenever a link layer wants to send data packets «as soon as possible.» There is only one type of arbitration on a 1394 cable-environment: fair arbitration.

• «Fair» arbitration guarantees equal access to the bus for all participating nodes. This prevents nodes that have a higher natural priority from dominating traffic on the bus.

Asynchronous transmit refers to the use of the AT FIFO interface.

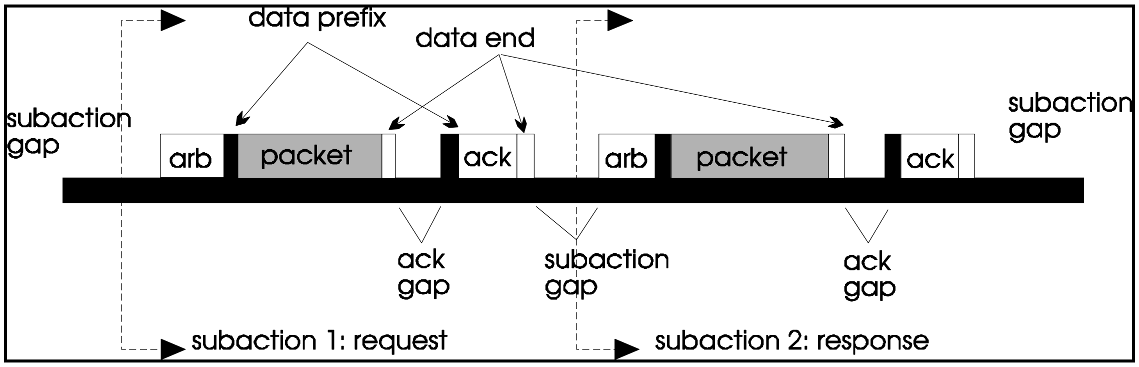


Figure 5: Subaction gaps are idle periods separating subactions.

All asynchronous subactions are normally separated by periods of idle bus called «subaction gaps» as shown in figure 1-2. The «Ack» is the time between reporting the end of a packet (data_end) and requiring that an acknowledging link layer respond with an arbitration request. The data prefix time is the remaining time after speed sampling before clocked data starts.

1.3.1.2 Isochronous Subaction

Isochronous arbitration is adequate for nodes that do not require a guaranteed high bandwidth or low latency or a precise timing reference. However, data such as that related to digital sound or instrumentation are more efficiently handle with isochronous arbitration. The cycle master tries to transmit a special timing request called a «cycle start" at specific intervals set by cycle synchronous source of 8 kHz, or 125µSec. Nodes sending isochronous data respond to cycle starts by immediately arbitrating for the bus without waiting for a subaction gap and sending an isochronous packet when arbitration succeeds.

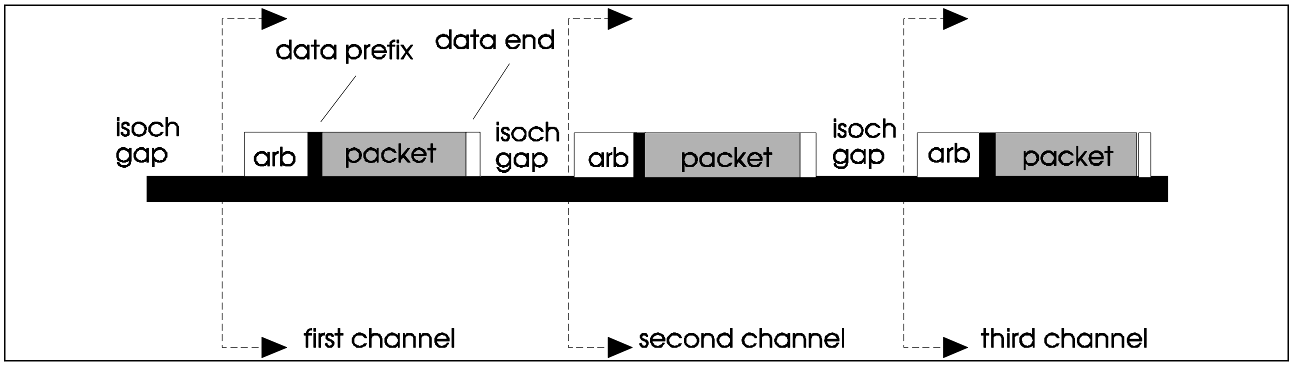


Figure 6: Isochronous gaps are periods between isochronous subactions.

Isochronous subactions are separated by periods of idle bus called «isoch gap». The nominal cycle period is 125µSec. Each node has its own 32-bit cycle timer register.

1.3.2 1394 parameters

There are parameters that need to be set in order to properly read and write 1394 packets. These parameters are the following:

- the 1394 node number : This is the node number that your card is being attributed on the bus. This number is re-configurable and can be changed at any time. The node number must be between 0 and 63 and there must not be two identical node number on the same bus.

- The 1394 bus number: Many different buses can be connected together on the same system. Each bus must have a different number and this number must be between 0 and 1023.

- The 1394 response flag: This flag is used to tell the system if it must send a 1394 read or write response for every read or write request that is being sent. When high performance real-time is required the response may not be sent. As a result, bandwidth will be higher, but reliability may decrease and 1394 standards won’t be followed.

1.4 VME Bus Overview

The master VME Interface allows a direct integration of the SD-VME-200-2 adapter card on any VME Bus system. The interface allows the card to become system controller, thus arbitrating bus grant to all the other cards on the system. The adapter card can also be setup as a VME master as well as a VME slave board, thus it can access other cards as well as be accessed by the other VME boards connected to the bus.

Addressing a memory map defined for the VME bus can access the VME bus (see Section 4: The VME bus interface for more details on this).

2. SYSTEM CONFIGURATION

2.1 SedNet Adapter Card Boot-Up Sequence

At boot up, the system will display the words SedNet on the LCD display followed by the software version number.

Then, a sequence of 5 tests is executed. These tests are as follow:

- Test 1 is the user SRAM test. If anything goes wrong during the test, the message: «ERR1» will show on the LCD display, which is located on the front panel of the card.

- Test 2 is the DRAM test. The LCD display will show the available amount of DRAM memory in the system. This can be 0 MB or 32 MB. The message «ERR2» will be displayed on the LCD if the test fails.

- Test 3 is the 1394 adapter test. This will test and initialize the physical and link layer of the 1394 protocol. «ERR3» will appear if the test goes wrong.

- Test 4 is the VME interface test. The message «ERR4» will be displayed if anything goes wrong with the test.

- Test 5 is the VME registers test. The message «ERR5» will be displayed when an error occurs during the test.

Note: see annex IV for a description of the error messages.

After all the tests are passed successfully, the LCD display will show the 1394 bus and node number of the SD-VME-200-2 card. When one 1394 connector or more are connected to the board, a vertical line will appear beside any of the last three digits of the LCD display. One line should appear for each connector successfully detected, showing the position of the connector being detected.

Example: If the 1394 bus number is 0x01 and the node number is 0x20, the LCD will display «0030» in hexadecimal. If there's a 1394 connector in the 1394 sockets of the card, a vertical bar will be displayed near the second digit of the LCD display.

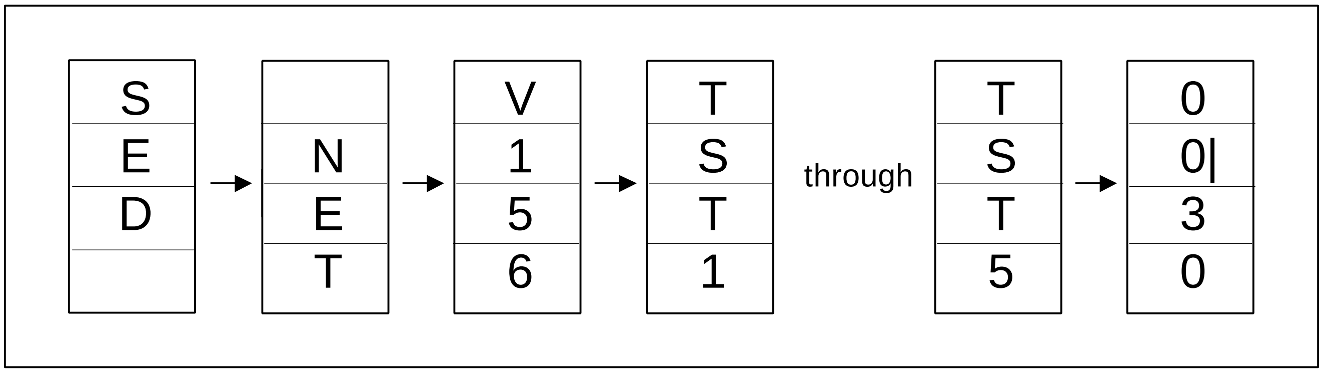


Figure 7: Example of a boot-up sequence on the LCD display

2.2 SedNet Adapter Card Jumper Configuration

There are two jumpers on the card, one for power supply capabilities by the 1394 connectors when the VME is not powered on (J1) and one for VME bus controller capability (J2). These jumpers are located as follow on the adapter card:

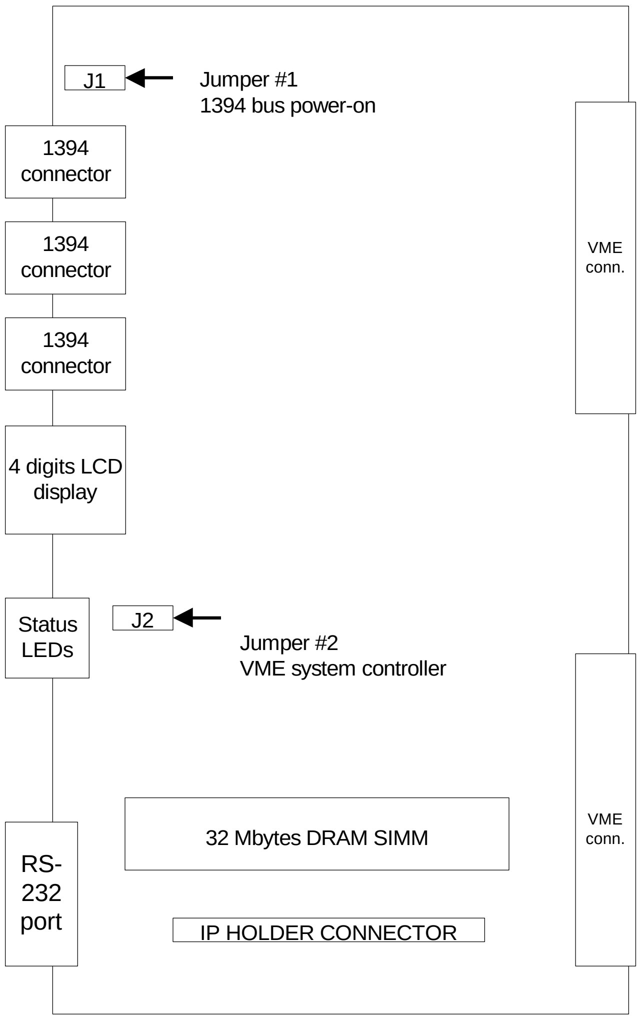


Figure 8: Position of the jumpers on the card

J1 is used to indicate if the PHY (1394 physical layer) can be powered by the 1394 bus. When the jumper is connected, the PHY can receive power from the 1394 bus even if the VME bus is off.

When jumper J2 is connected, the board can act as VME system controller and can perform VME bus arbitration.

2.3 SedNet Adapter Card Parameters Configuration

In order to have a fully functional system, some parameters need to be setup. Normally, all the parameters should be setup at system delivery. If for any reason you want to change some of the parameters, here's how to do it, with the description of each parameter. (for a quick reference of the parameters and how to change them see Annex I and Annex II at the end of this document).

2.3.1 Parameters list

The important parameters are:

- Serial Number
- Version Number
- Bus Node and Bus Number
- Response Flag
- RS-232 port configuration flag
- VME Address Size
- VME Data Size
- VME Bus Mapping

Serial Number:

This parameter is used to indicate the serial number of an adapter card. All adapter cards have distinct serial number. That's the only way to distinguish one card from another. This parameter is usually written once and never modified after.

Version Number:

This parameter is used to give the software and hardware version of the card. It's mostly used when a new version of the software is given, in order to identify the current version on the card. Here's how the version can be read :

31 Hardware Version 16	15 Software Version 0

Bus Node and Bus Number:

The Node Number is the number of the 1394 node to identify your card in the 1394 network configuration. The bus number is used when there are more than one 1394 buses for example, there could be two node number 2, one on bus 1 and one on bus 2. By default, both the bus node and the bus number are set at 0. The bus number can take values between 0 and 1023 and the node number can take values between 0 and 63.

31 Bus Number 22	21 Bus Node 16	15 Reserved 0

Response Flag:

This parameter tells the system if it must send an acknowledgement for every read or write request that is being sent. When the acknowledge is not send more packets can be send in a shorter amount of time, but there's no way to verify if the packet as been send the proper way.

These are the possible value for the parameter:

Value	Status
0	No Response
1	Send a Response

Table 2: Configuration values for the Response flag

RS-232 Configuration Flag:

This parameter tells the system if the RS-232 port is being used as a regular RS-232 port for data transfer or as a debugger for the system. In the later case, the port will be connected to a terminal in order to monitor system activity or to configure and test the system.

These are the possible value for the parameter:

Value	Mode
0	monitor mode
1	regular RS-232 port

Table 3: Configuration values for the RS-232 port

VME Address Size:

This parameter is the size of the addresses when making an access to the VME bus. The values for this parameter are 16, 24 and 32 bit addresses (for more information on how to setup the address size, see section 2.3.1 on the VME bus settings). Make sure that the address size are the same for all the cards on your system or incompatibilities could occur at address decoding.

VME Data Size:

This parameter is the size of the data on the VME bus. The values for the parameter are 16 and 32 bits (see section 2.3.1 for more information concerning the setup of the data size)

VME Bus Mapping:

These parameters will do two things:

-Map your card on the VME Bus
-Map the memory area that can be accessed on your card

(see section 2.3.2 for more information about the VME Bus Mapping)

2.3.2 How to change the parameters

It is possible to change a parameter by writing to its assigned memory space. Unfortunately, the update will not be permanent. If the new values for the parameters are written to the FLASH memory, they will be kept permanently (until they are changed again) and will be updated automatically after the operation.

Here is what you need to do to change the parameters to FLASH memory :

Action	Address	Data
STEP 1: Write the new parameter values to SRAM	Parameters addresses	New parameter values
STEP 2 : Write the update parameter semaphore to start copying to FLASH memory	0x000AFF88	0x00000001

Table 4: How to update the system parameters

2.3.3 Parameters addresses

PARAMETER	ADDRESS
SERIAL_NUMBER	0x000AFF00
VERSION_NUMBER	0x000AFF04
BUS_NODE	0x000AFF08
RESPONSE_FLAG	0x000AFF0C
ISO1_CHANNEL_NUM	0x000AFF10
ISO2_CHANNEL_NUM	0x000AFF14
CS5_ASIZ (used for VME configuration)	0x000AFF18
SLAV_ADD (used for VME mapping)	0x000AFF1C
SLAV_ADD_MSK (used for VME mapping)	0x000AFF20
VME_ADD 7(used for VME mapping)	0x000AFF24
VME_ADD_MSK (used for VME mapping)	0x000AFF28
1394 Packet length and packet address high (only used when accessing 1394)	0x000AFFAC
1394 Packet address low (only used when accessing 1394)	0x000AFFB0
1394 Response packet address high (only used when accessing 1394)	0x00AAFFB4
1394 Response packet address low (only used when accessing 1394)	0x00AAFFB8
RS-232 Port Configuration	0x00AAFFBC

Table 5: Addresses for the system parameters

2.4 VME System Configuration

2.4.1 VME bus settings (Address Size and Data Size)

The VME bus settings are the settings that determine the Address Size and Data Size. The ADDRESS of the parameter in memory is 0x000AFF18. You will need to write one of the following values to that address in order to configure your VME bus:

Value	Data Size	Address Size
0x01000000	16 bits	32 bits
0x02000000	16 bits	16 bits
0x03000000	16 bits	24 bits
0x05000000	32 bits	32 bits (default value)
0x06000000	32 bits	16 bits
0x07000000	32 bits	24 bits

Table 6: Configuration of the VME bus size and address size

Once the value is written, you will need to activate the changes that you made. In order to do this, you will need to do one of the three alternatives:

	Action	Address	Value
Alternative #1	STEP 1 : write to the VME update semaphore (this will only activate the change in the parameters, but they won't be kept in FLASH)	0x000AFF98	0x00000001
Alternative #2	STEP 1 : write only the VME parameters to FLASH (this will also activate the changes)	0x000AFF9C	0x00000001
Alternative #3	STEP 1 : do as in section 2.3.2 to change the parameters to FLASH	See section 2.3.2	See section 2.3.2

Table 7: How to activate the VME parameter changes

Note: it is recommended that you save the changes to FLASH memory only once all the VME parameters have been entered.

2.4.2 VME mapping of the 1394-VME master adapter

In order to be able to be recognized on VME memory area, the card's VME parameters must be properly set. The settings for the card will do two things:

- it will map the adapter card on the VME bus, giving it an exclusive address on the bus.
- it will map the VME space that can be addressed from the adapter card.

2.4.2.1 How to map your card on the VME bus

Here are all the steps you need to follow to map your adapter card on the VME bus:

Action	Address	Value
STEP 1 : Write the address of the card on the VME bus.	0x000AFF24	The card address on the VME bus
STEP 2 : Write the address mask that will indicate the memory zone occupied on the bus by the card.	0x000AFF28	The address mask suggested is 0x03FFFFFF
STEP 3 : Save the changes to FLASH memory (optional)	0x000AFF9C	0x00000001

Table 8: Mapping the card's base address on the VME bus

Note: it is recommended that you save the changes to FLASH memory only once all the VME parameters have been entered.

2.4.2.2 How to map the memory area that can be accessed on your card from the VME bus

When your card is on the VME bus you may only want a part of its memory mapping to be available to other cards. In order to do that, you will need to setup your VME memory area. Here's how to do it :

Action	Address	Value
STEP 1 : Write the base address of the memory area that will be addressed on the card	0x000AFF1C	Suggested address : 0x00000000
STEP 2 : Write the size of the memory area that will be accessed on the card from the bus.	0x000AFF20	Suggested size : 0x03FFFFFF
STEP 3 : Save the changes to FLASH memory (optional)	0x000AFF9C	0x00000001

Table 9: Mapping the card's memory area on the VME bus

Note: it is recommended that you save the changes to FLASH memory only once all the VME parameters have been entered.

Example:

> We want to map this card: its address on the bus will be 0x08000000 on the VME bus and the first 64 Mbytes will be available (we can address 0x00000000 to 0x03FFFFFF on the card). We also want the card to have 32 bits data and addresses.

1- First write 0x05000000 at memory location 0x000AFF18 to have 32 bits address and data size.

2- Write 0x00000000 at location 0x000AFF1C, which is the slave address register. The card can now be accesses from the VME bus at address 0x0.

3- We also have to set the slave address mask register (location 0x000AFF20) to 0x03FFFFFF, so we can access 64 Mbytes of the card from the VME.

4- Then, we write 0x08000000 at location 0x000AFF24 to set the card address to 0x08000000.

5- The VME address mask register (address 0x000AFF28) will be set to 0x03FFFFFF.

When all the parameters have been set, writing to the VME update semaphore can perform activation of the parameters. To write the semaphore, 0x1 has to be written at address 0x000AFF9C.

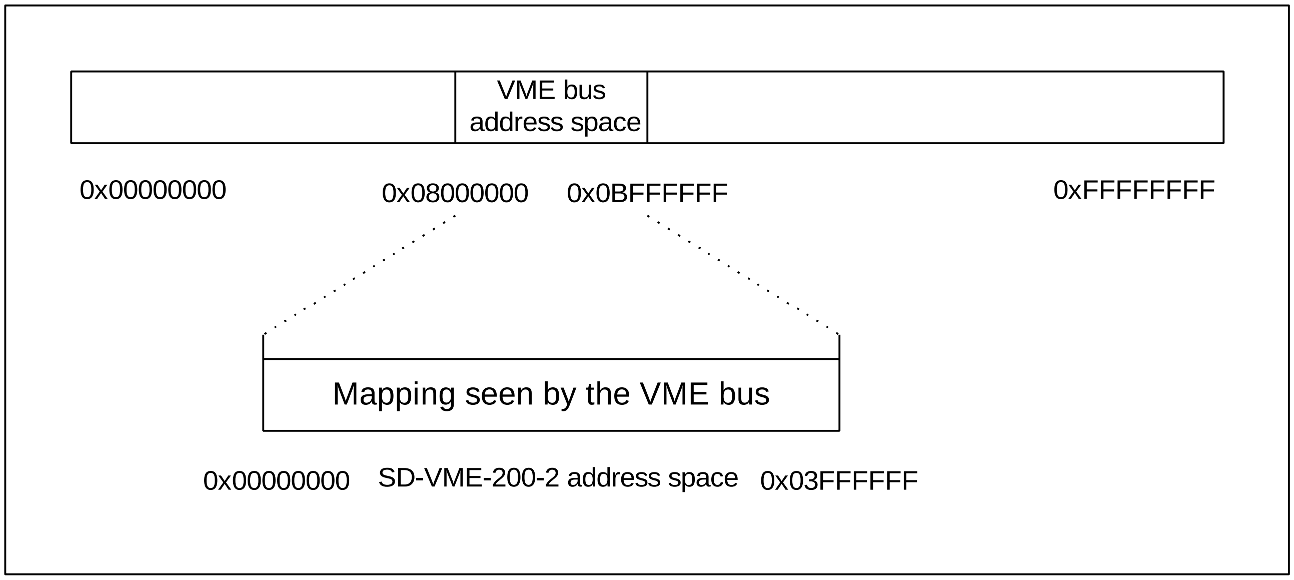


Figure 9: An example of a SD-VME-200-2 VME mapping

Note: The valid addresses to reach the SD-VME-200-2 from another card on the VME bus are from 0x08000000 to 0x0BFFFFFF. (base address: 0x08000000, offset range: 0x00000000 to 0x03FFFFFF)

2.4.3 System controller

When the 1394-VME card is set as a system controller, it can perform bus arbitration on the VME. The position of the jumper J2 set the 1394-VME card as system controller.

This jumper must be put at location J2 on the card. The system controller card has to be in the slot 0 of the VME bus.

Note: only one board on the VME bus should have his VME system controller jumper set.

3. USING THE SD-VME-200-2 AS A BRIDGE BETWEEN VME BUSES.

There are three easy steps to follow when you want to use the 1394 interface as a data exchange channel from the VME bus:

1) Write the 1394 request to memory. All the information concerning how to write a 1394 packets are located in sections 3.1, 3.2, 3.3 and 3.4.

2) When the request is written, write to the 1394 semaphore, so the data transfer will be started by DMA.

3) Once the transfer is started, wait for the write or read response that contains the requested data.

Explanation: This process transmits a 1394 Read or Write request packet from another VME card, thus allowing to Read or Write data anywhere on the 1394 bus from the bridge card that is being accessed by all other non 1394 cards. These cards just have to transmit Read and Write request to the specified VME Bridge in order to get any data on the other VME buses. See Figure 3-1 for more details.

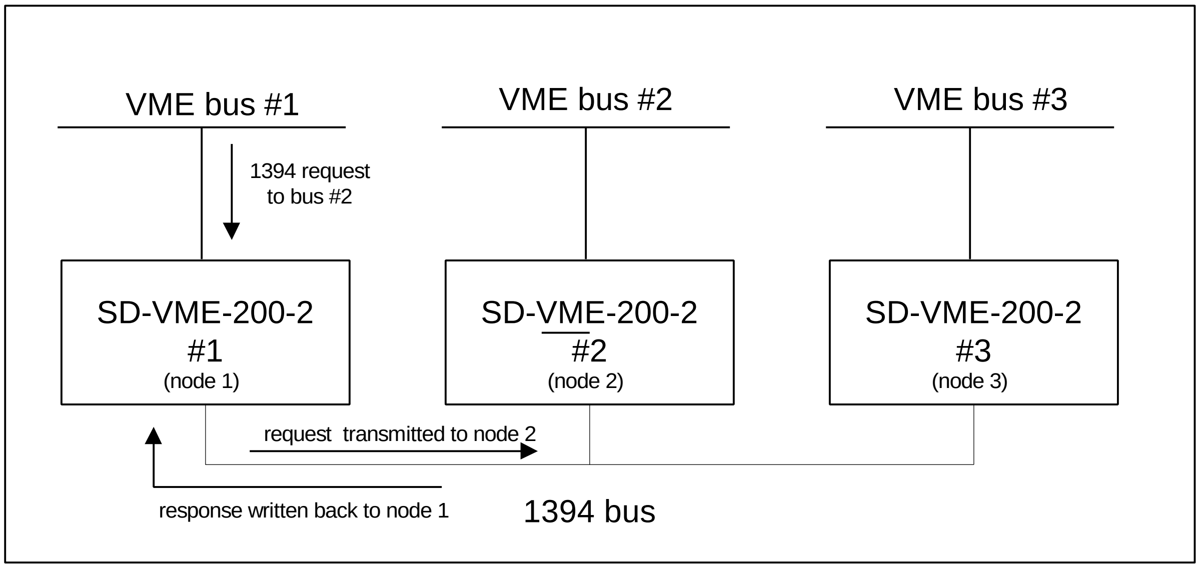


Figure 10: Example of a 1394 bridging request

3.1 Writing 1394 quadlet

3.1.1 The Write Quadlet Request

This first 1394 operation writes a single data quadlet to a specified destination address. The first quadlet contains control information, the second, the destination (node and bus number).

31 24	23 16		15 14 8		7 0	
0		spd	tLabel	0	tCode	0
Destination ID			0000			
Destination Offset Low						
Quadlet data						

Here's a description of the fields contained in a write request :

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
TLabel	Transaction label, unique tag for each outstanding transaction between two nodes.
tCode	tCode is the transaction code for this packet = 00
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address of this packet.
Destination Offset Low	This field address represents the destination nodes address space (Address must be quadlet aligned).
Quadlet Data	This field holds the data to be transferred.

Table 10: Description of the fields for the write quadlet request

Example:

- Write the quadlet equal to 0xaaaaaaaa (32 bits) in the DRAM of the SD-VME-200 at the address 0x8000000 at 200Mb/s.
- write request done by the node 0 and the bus 0, to the node 2 and the bus 0

31 24	23 16	15 14 8	7 0
0x00010000			
0x00020000			
0x08000000			
0xaaaaaaaa			

3.1.2 The Write Quadlet Response

This packet is received after sending a write quadlet request or a write data block request. This packet won't be sent if the flag RESPONSE_1394 is reset.

31 24	23	16	15 14	8	7	0
Destination ID			TLabel	0	tCode	0
Source ID			0000			
Destination Offset Low						
000		spd	000			ackSent

Here's a short description of each of the response's fields:

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
tCode	tCode is the transaction code for this packet = 0010
Source ID	This is the node ID of the sender of this packet
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address to which this packet is being sent.
Destination Low	This field address represents the destination nodes address space (Address must be quadlet aligned, and Upper four bits of the destination Offset High is used as the response code).

Table 11: Description of the fields for the write quadlet response

3.2 Reading 1394 quadlet

3.2.1 The Read Quadlet Request

This packet type requests a data quadlet form a specified destination address.

31 24	23	16	15 14	8	7	0
0		spd	tLabel	0	tCode	0
Destination ID			0000			
Destination Offset Low						

Here's a short description a the fields for a read request:

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
tCode	tCode is the transaction code for this packet = 0100
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address of this packet.
Destination Offset Low	This field address represents the destination nodes address space (Address must be quadlet aligned).

Table 12: Description of the fields for the read quadlet request

<u>Example</u>:

- read a quadlet in the DRAM of the SD-VME-200 at the address 0x8000000 at 200Mb/s.
- read request done by the node 0 and the bus 0, to the node 2 and the bus 0

31 24	23 16	15 14 8	7 0
0x00010040			
0x00020000			
0x08000000			

3.2.2 The Read Quadlet Response

This packet is received after sending a read quadlet request. This packet won't be sent if the flag RESPONSE_1394 is reset. This packet type sends a single data quadlet from a specified destination address.

31 24	23 16		15 14 8		7	0
Destination ID			tLabel	0	tCode	0
Source ID			0000			
Destination Offset Low						
Quadlet data						
000		spd	000			ackSent

Here's a brief description of the fields contained in a read quadlet response.

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
Tcode	tCode is the transaction code for this packet = 0110
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address of this packet.
Destination Offset Low	This field address represents the destination nodes address space (Address must be quadlet aligned).
Quadlet Data	This field holds the data to be transferred.

Table 13: Description of the fields for the read quadlet response

3.3 Writing 1394 Blocks

3.3.1 The Write Block Request

This packet contains a block data to write at a specified destination address.

31 24	23 16		15 14 8		7	0
0		spd	TLabel	0	tCode	0
Destination ID			0000			
Destination Offset Low						
Data Length			0000			
Block Data						

Here’s a description of the different fields in a write block request :

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
Tlabel	Transaction label, unique tag for each outstanding transaction between two nodes.
Tcode	TCode is the transaction code for this packet = 01
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address to which this packet is being sent.
Destination Low	This field address represents the destination nodes address space (Address must be quadlet aligned, and Upper four bits of the destination Offset High is used as the response code).
Data Length	The number of bytes to transmitted in the packet
Block Data	This field holds the data to be transferred.

Table 14: Description of the fields for the write block request

Example:

- Write 4 quadlets equal to 0xaaaaaaaa (4 x 32 bits) in the DRAM of the SD-VME-200 at the address 0x9000000 at 200Mb/s.
- write request done by the node 0 and the bus 0, to the node 2 and the bus 0

31 24	23 16	15 14 8	7 0
0x00010010			
0x00020000			
0x09000000			
0x00040000			
0xaaaaaaaa 0xaaaaaaaa 0xaaaaaaaa 0xaaaaaaaa			

3.3.2 The Write Block Response

This packet is received after sending a write data block request. This packet won’t be sent if the flag RESPONSE_1394 is reset.

31 24	23 16	15 14	8	7	0
Destination ID		TLabel	0	tCode	0
Source ID		0000			
Destination Offset Low					
000	spd	000			ackSent

As usual, here's a description of the fields contained in the write block response :

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
Tcode	tCode is the transaction code for this packet = 0010
Source ID	Thisis the node ID of the sender of this packet
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address to which this packet is being sent.
Destination Low	This field address represents the destination nodes address space (Address must be quadlet aligned, and Upper four bits of the destination Offset High is used as the response code).

Table 15: Description of the fields for the write block response

3.4 Reading 1394 Blocks

3.4.1 Read Block Request

This packet type requests a data block at specified destination address.

31 24	23	16	15 14	8	7	0
0		spd	tLabel	0	tCode	0
Destination ID			0000			
Destination Offset Low						
DataLength			0000			

Here's the description of the fields needed to write a block read request :

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
TCode	tCode is the transaction code for this packet = 0101
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address to which this packet is being sent.
Destination Low	This field address represents the destination nodes address space (Address must be quadlet aligned, and Upper four bits of the destination Offset High is used as the response code).
Data Length	The number of bytes to transmitted in the packet

Table 16: Description of the fields for the read block request

Example:

- read 128 quadlets in the DRAM of the SD-VME-200 at the address 0x8200000 at 200Mb/s
- read block request done by the node 0 and the bus 0, to the node 2 and the bus 0

31 24	23 16	15 14 8	7 0
0x00010050			
0x00020000			
0x08200000			
0x01000000			

3.4.2 The Read Block Response

This packet type sends a block data form a specified destination address.

31 24	23 16	15 14	8	7	0
Destination ID		Tlabel	0	tCode	0
Source ID		0000			
Destination Offset Low					
Block data (if any)					
000	spd	000			ackSent

Here's a description of the read block response fields:

FIELD NAME	DESCRIPTION
Spd	This field indicates the speed at which this packet is to be sent. 00 = 100 Mb/s, 01 = 200 Mb/s, and 10 = 400 Mb/s 11 is undefined for this implementation
TCode	TCode is the transaction code for this packet = 0111
Destination ID	This is the concatenation of the 10-bit bus number and the 6-bit node number that forms the destination nodes address to which this packet is being sent.
Destination Offset Low	This field address represents the destination nodes address space (Address must be quadlet aligned, and Upper four bits of the destination Offset High is used as the response code).
Data Length	The number of bytes to transmitted in the packet
Block Data	This field holds the data to be transferred.

Table 17: Description of the fields for the read block response

3.5 Activating the data transfer

Here are the steps you need to follow to send the 1394 packet:

***Note: you need to take control of the 1394 interface before you write the 1394 request. The 1394 request can be written at the address you want in user SRAM, in DRAM or at a location on the VME bus

Action	Address	Value
STEP 1 : Wait until the data at address 0x000AFFA8 is 0x0000. (this means that the 1394 interface is not busy)	0x000AFFA8	0x00000000
STEP 2 : Write 0x01 to the 1394 semaphore to take its control	0x000AFFA8	0x00000001
STEP 3 : Write the 1394 request packet as explained in sections 3.1 to 3.4	Your selected address	Your 1394 Request Packet
STEP 4 : Write the address of the 1394 packet that you wrote	0x000AFFB0	Address of the 1394 request packet (as in step 3)
STEP 5 : Write the length of the packet in the 16 most significant bits at address 0x000AFFAC.	0x000AFFAC	|32 length 16|15 origin 0|
STEP 6 : Write the origin of the packet in the 16 less significant bits of address 0x000AFFAC. (write a 0 if on the card and 0x1A if on the VME bus)	0x000AFFAC	|32 length 16|15 origin 0|
STEP 7 : Write the address destination offset low of the response packet	0x000AFFB8	Address of the 1394 response packet to be received
STEP 8 : Write the address destination offset high of the response packet. (0x0 if on the card, 0x1A if on the VME bus)	0x000AFFB4	Address offset high of the response packet to be received (0 is on the card, 1A on the VME)
STEP 9 : Start the transfer by writing a 0x02 at address 0x000AFFA8	0x000AFFA8	0x00000002
STEP 10 : Wait until the data at address 0x000AFFA8 is 0x0 (this means that the transfer is over)	0x000AFFA8	0x000000000
STEP 11 : Pick up the 1394 response packet at the address specified in STEP 6 and STEP 7	See STEP 6 and 7	1394 Response Packet

Table 18: Steps to follow to activate 1394 packet transmission

Example :

Picking up a packet located on at address 0x000A0000, with a packet length of 64 bytes. The write response will be looked for at address 0x000C0000 on the board after the request is executed.

1- Wait while the 1394-VME master adapter is available (when data at 0x000AFFA8 is equal to 0x0).

2- Request the 1394 bridge by writing 0x1 at address 0x000AFFA8.

3- Write the 1394 request packet at address 0x000A0000 on the SD-VME-200-2 board.

4- Write 0x000A0000 at address 0x000AFFB0, to indicate where the packet has to be picked up.

5- Write 0x00400000 at address 0x000AFFAC.

    0040 is the length and 0000 means that the request is directly on the card.

6- Write 0x00000000 at address 0x000AFFB4. (because the acknowledge is not located on the VME, otherwise, it would have been 0x1A)

7- Write 0x000C0000 at address 0x000AFFB8 (this is the acknowledge address).

8- Write 0x00000002 at address 0x000AFFA8 this will start packet transfer.

9- Wait for the response packet at address 0x000C0000 on the board until the value at 0x000AFFA8 is 0x0.

4. THE RS-232 SERIAL PORT (NOT AVAILABLE)

4.1 Overview of the RS232 Port

The RS232 port on the SD-VME-200-2 card serves many purposes:

- It can be used as an RS232 port to connect to a printer, a modem or any serial communication device.
- It can be connected to a terminal in order to monitor 1394 packets on the card and VME access to the card. Memory blocks can be read and some software functions (semaphores) can be used with this option.
- It can also be used as a debugging tool to read and write memory blocks, configure the system parameters, update the card's firmware or perform software reset of the system.

4.1.1 RS232 port mode

In RS232 mode, the port is used as an input/output port for serial communication. No special software or hardware is required to communicate with the port and any RS232 device can be accessed via the serial port. The only settings that may need to be changed are the communication parameters such as baud rate, data bits, parity, start and stop bits.

4.1.2 Monitor mode

In monitor mode, the port acts as a window where the user can see the packets being sent from and to the system. It can also be used to read the system's memory, but no writes are allowed since this could put the system in an unstable state. Some of the software functions (semaphores) can be used from this option and the system configuration parameters are always available to the user.

4.1.3 Debug mode

In debug mode, no software is running on the card. Memory read and write can be performed as well as memory block fills. The system configuration can be accessed as well as being changed from this option. New version of the firmware can be uploaded to the FLASH (software upgrades become very easy) and 1394 accesses can be made for testing purposes.

5. HOW TO UPDATE YOUR CARD’S FIRMWARE

5.1 Updates from the RS232 Port (not available)

5.2 Updates with our application software.

Our application software is a WinNT program used in the MS-DOS environment. It should be included in your package when you buy a SD-VME-200-2 card with a SD-PCI-200 adapter card. The program name is updatevme and must be called in a MS-DOS window on the host computer of the SD-PCI-200-2 card. By calling the program with the “-h” switch, you can get the online help.

Here’s what the online help looks like:

```
-SD-VME-200-2---PROGRAM-FLASH-UPDATE----------Version-1.3----

  UpdateVME [-n0..60] [-N0..60]

  -n  node number of the board to update  (default is 10)
  -N  node number of the PCI board        (default is 26)
  -h  help menu

----------------------------------------------------------------
```

During execution, the application will ask for the firmware filename, this is a “.dwn” file that we provided you with when an update version of the firmware is released. This file and the application file must be in the same directory.

When performing the update, the LCD display will show the percentage of the update that’s been performed. After the update, the LCD display should show the 1394 bus and node number again, otherwise, an error occurred while updating the firmware.

Example: to update node 32 on the bus, the command is:

updatevme –n32.

Note: After updating the firmware, a RESET of the card must be performed in order to activate the new firmware. (See ANNEX II for the address of the semaphore that resets the card)

⚠ Caution : Performing a FLASH update may cause loss of all data in user SRAM because of card reboot process.

6. MISCELLEANOUS FUNCTIONALITIES AND FEATURES

6.1 Writing your own parameters to user FLASH memory.

You can write your own parameters and values to FLASH. In order to do that, you must follow the procedure below:

Action	Address	Value
STEP 1 : Write your data in a memory zone other than the FLASH area. (The data will be copied from this zone to the FLASH)	Address of the memory zone where the data are located	Data Value
STEP 2 : Write the address offset low of the data block to copy to FLASH in 0x000AFFC4. (same address as above)	0x000AFFC4	Address of the data block to copy
STEP 3 : Write the length in bytes of the data block to copy to FLASH in the 16 most significant bits at address 0x000AFFC0.	0x000AFFC0	|32 length 16|15 origin 0|
STEP 4 : Write the address offset high of the data in the 16 less significant bits of address 0x000AFFC0. (write a 0 if on the card and 0x1A if on the VME bus)	0x000AFFC0	|32 length 16|15 origin 0|
STEP 5 : Write the address of the FLASH memory block where the data will be copied in 0x000AFFC8.	0x000AFFC8	Address of the FLASH memory block
STEP 6 : Start the update FLASH semaphore by writing a 1.	0x000AFFBC	0x000000001

When updating the FLASH, the system should display the percentage of the update being perfomed and return to bus/node display when the update is over.

⚠ Caution : Performing a FLASH update may cause loss of all data in user SRAM because of card reboot process.

6.2 Knowing how many 1394 adapters are connected to the 1394 bus with self-id packets.

By reading at address 0x000AFD00, you can read a FIFO with a certain amount of self-id packets. These packets tell the SD-VME-200-2 card how many cards are connected to the bus and what is the exact configuration (topology) of the 1394 bus.

These packets are 64 bits packets that can be interpreted by reading section 4.3.4.1 of the IEEE Std 1394-1995 (IEEE Standard for a High Performance Serial Bus).

They are written one after another in the memory space starting at 0x000AFD00 and can be read until an 32 bit word containing 0x00000000 is being read, this means that no more self-id packets can be found in the FIFO.

6.3 Performing communication tests with the SD-VME-200-2 with the application software.

Included in our software library provided with the SD-VME-200-2 card is a program called commvme that can be executed in an MS-DOS window under WinNT.

Executing the program without a parameter or with the “-h” switch will display the help screen. Here’s what it looks like :

```
-SD-VME-200-2---1394-COMMUNICATION-TEST---------Version-1.4----

  CommVME [-s/-d] [-axxxxxxxx] [-n0..60] [-N0..60] [-V] [-i] [-p] [-h]

  -s  Complete SRAM test                  (default is non valid)
  -d  Complete DRAM test                  (default is non valid)
  -n  node number of the tested board     (default is 10)
  -b  bus number of the tested board      (default is 0)
  -a  VME offset of the tested board      (default is 0x00000000)
  -N  node number of the PCI board        (default is 26)
  -V  speed (100Mbs)                      (default is 200Mb/s)
  -i  number of test loop                 (default is 1)
  -p  no printout option                  (default is ON)
  -h  help menu

---------------------------------------------------------------
```

example : to perform a SRAM test on node 32 at a speed of 100Mbps with 1000 iterations and without any printouts, the command would be:

comvme –n32 –V –i1000 –p –s.

6.4 Updating the SD-VME-200-2 parameters with the application software.

Included in our software library provided with the SD-VME-200-2 card is a program called showvme that can be executed in an MS-DOS window under WinNT. Executing the program without a parameter or with the “-h” switch will display the help screen. Here’s what it looks like :

```
-SD-VME-200-2---SETUP-PARAMETERS-----------------Version-1.2----

  ShowVME [-r/c] [-V] [-n0..60] [N0..60] [-b] [-X] [-h]

  -r  read parameters
  -c  change parameters
  -n  node number of the tested board     (default is 10)
  -b  bus number of the tested board      (default is 0)
  -N  node number of the PCI board        (default is 26)
  -V  speed (100Mbs)                      (default is 200Mb/s)
  -X  research all SD-VME-200-2 connected on the 1394 bus
  -h  help menu

---------------------------------------------------------------
```

Example: to see the parameters for n32, the command is:

showvme –r –n32

ANNEX I: SYSTEM PARAMETERS

These are the system parameters

ADDRESS	PARAMETER	DESCRIPTION
0x000AFF00	SERIAL_NUMBER	Serial number of the card
0x000AFF04	VERSION_NUMBER	Version number for the software on the card
0x000AFF08	BUS_NODE	1394 node and bus number
0x000AFF0C	RESPONSE_FLAG	1394 response flag (a response is sent if the flag is 0x1)
0x000AFF10	ISO1_CHANNEL_NUM	Isochronous1 channel number
0x000AFF14	ISO2_CHANNEL_NUM	Isochronous2 channel number
0x000AFF18	CS5_ASIZ	Address size and data size of the VME
0x000AFF1C	SLAV_ADD	Base address of the memory map that can be accessed on the card
0x000AFF20	SLAV_ADD_MSK	Mask for the address on the card
0x000AFF24	VME_ADD	Address of the card on the VME bus
0x000AFF28	VME_ADD_MSK	Mask for the address on the VME bus
0x000AFFAC	1394 Packet length and packet address high (only used when accessing 1394 with the 1394 semaphore)	The first 16 bits are used for the length of the packets in byte. The second 16 bits are used for the packet address high. Note: if the address given is 0, the packet is on the board, if it's 0x1A, the packet is located on the VME bus.
0x000AFFB0	1394 Packet address low (only used when accessing 1394 with the 1394 semaphore)	The 32 bits are used as the destination address. If the packet is on the VME bus, the address has to be decoded by the VME interface.
0x00AAFFB4	1394 Response packet address high (only used when accessing 1394 with the 1394 semaphore)	The 1394 interface must send a response packet when the packet is delivered. This is the high part of the address of the response that must be sent, 0 means that the response will be on the board, 0x1A means it will be on the VME bus.
0x00AAFFB8	1394 Response packet address low (only used when accessing 1394 with the semaphore)	This is the address of the 1394 response packet.
0x00AAFFBC	RS-232 Port Configuration (not used for now)	This is used to configure the RS-232 port as a regular RS-232 port or as a system monitor-debugger. 0 => the port is set as a system-monitor 1 => the port is set as a regular RS-232 port

ANNEX II: SEMAPHORES

- What's a semaphore?

A semaphore is a message sent to the application telling that an action needs to be performed or has been performed. When a value is written at a certain address, the CPU will know that it must start some actions. Once the action is completed, the CPU usually answers back by telling that the action is done.

This is a table with all the semaphores mapped on the card. Most of the time, writing a 0x01 to the address will start the semaphore and a 0x00 will be written when the task is completed.

ADDRESS	SEMAPHOR	DESCRIPTION
0x000AFF80	RESET	This semaphore causes a software reset of the board being reset.
0x000AFF84	VME Reset	This semaphore causes a VME reset, which is a hardware-reset card. Note: when using this semaphore, the whole VME bus will be reset.
0x000AFF88	Save parameters in FLASH	This semaphore takes the parameters in the SRAM and writes them to the user FLASH. This is the only way to change the parameters in the FLASH since it can't be accessed directly by the user. The values in the FLASH are the one given to the system at system boot-up.
0x000AFF8C	Restore default parameters	This semaphore takes the default parameters in the default flash section and copies them in the SRAM. This is to be used only when you want to recover the system to it's default values.
0x000AFF90	Update parameters	This semaphore takes the parameters in the user FLASH area and copies them to the SRAM.
0x000AFF94	Bus Error	This semaphore is used internally to monitor bus errors encountered. (reserved to the system)
0x000AFF98	Update VME registers	This semaphore takes the VME registers parameters in the SRAM area and initializes the VME registers to the required values.
0x000AFF9C	Save VME value to FLASH	This semaphore takes the VME registers parameters in the SRAM and writes them to the user FLASH area for future boot up of the system.
0x000AFFA0	Update VME value to SRAM	This semaphore takes the VME registers value from the user FLASH area, copies them in the SRAM and initializes the VME registers.
0x000AFFA4	Restore VME default value	This semaphore takes the default VME registers value, copy them value to SRAM and initialize the VME registers.
0x000AFFA8	VME to 1394	This semaphore is used as a request for 1394 packet transmission from the VME, when the card is used as a bridge for 1394 packets. It also monitors the state of the 1394 devices.

ANNEX III: MEMORY MAPPING OF THE BOARD

This is the general memory mapping of the card. It contains all the memory location available from the card:

ADDRESS	MEMORY OR I/O	DESCRIPTION	Permissions
0x00000000 to 0x0003FFFF	Flash Memory (reserved for firmware)	256K of reserved flash space (512K, 8-bit)	Read Only
0x00040000 to 0x0007FFFF	Flash Memory	256K of user flash space (512K, 8-bit)	R/W
0x00080000 to 0x000BFFFF	Static Memory Bank 0	Static RAM (256K, 32-bit) (program SRAM area)	Read Only
0x000C0000 to 0x000FFFFF	Static Memory Bank 1	Static RAM (256K, 32-bit) (user SRAM area)	R/W
0x00100000	1394 Link Layer (TSB12C01)	This controller transmits and receives correctly formatted 1394 packet	Read Only
0x00100100	1394 state machine		
0x00100200	CS-VIC068A	VME Bus Interface	
0x00100300	VME registers		
0x01000000 to 0x01FFFFFF	DRAM SIMM 16M – Bank 0	72-PIN SIMM DRAM (16M – 32-bit)	R/W
0x02000000 to 0x02FFFFFF	DRAM SIMM 16 M – Bank 1	72-PIN SIMM DRAM (16M – 32 bit) (32M option only)	
0x03000000	IP MODULE 1	IP module 1 for single size module	
0x03000080	IP MODULE 2	IP module 2 for single size module	
0x03000100	IP MODULE 3	IP module 3 for single size module	
0x03000180	IP MODULE 4	IP module 4 for single size module	

This is the memory mapping for the IP modules (not implemented yet):

	IP module 0	IP module 1	IP module 2	IP module 3
ID SEL (64 bytes)	0x03000000	0x03000080	0x03000100	0x03000180
INT SEL (64 bytes)	0x03000040	0x030000A0	0x03000140	0x030001A0
IO SEL (64 bytes)	0x03000200 to 0x0300027F	0x03000280 to 0x030002FF	0x03000300 to 0x0300037F	0x03000380 to 0x030003FF
ID MEM (512 Kbytes)	0x03100000 to 0x0317FFFF	0x03180000 to 0x031FFFFF	0x03200000 to 0x0327FFFF	0x03280000 to 0x032FFFFF

[1] ID SEL is the location that contains specific information about your IP modules (Vendor ID, Product ID, IP type, etc. see your IP documentation for more details).

This is the memory mapping for the 1394 link layer that transmits and receives the 1394 packets. It describes the TSB12C01 registers:

ADDRESS	REGISTER	DESCRIPTION
0x00100000	Version	Version and Revision register
0x00100004	Node Address	Node address and Transmitter acknowledge
0x00100008	Control	Control register
0x0010000C	Interrupt	Interrupt register
0x00100010	Interrupt Mask	Interrupt Mask register
0x00100014	Cycle Timer	Cycle-timer register
0x00100018	Isoch. Port Number	Isochronous Receive Port number register
0x0010001C	Reserved	
0x00100020	Diagnostics	Diagnostic Control and Status register
0x00100024	Phy Chip Select	Phy-Chip Access register
0x00100028	Phy Interface State	Phy-Interface State register
0x0010002C	Other State	State Value register (all other modules except Phy)
0x00100030	ATF Status	Asynchronous Transmit-FIFO Status register
0x00100034	ITF Status	Isochronous Transmit-FIFO Status register
0x00100038	Reserved	
0x0010003C	GRF Status	General Transmit-FIFO Status register
0x00100040	Reserved	

ANNEX IV: ERROR MESSAGE

Error messages may be displayed on your LCD display in order to let you know that something wrong happened with your system.

Some error may be minor errors, others may be major errors or even unrecoverable errors.

Here's a list of the most common errors that could happen on your system:

Error message	Description	What to do in case of error
ERR1	SRAM error	
ERR2	DRAM error	
ERR3	1394 initialization error	
ERR4	VME interface error	
ERR5	VME registers error	
ERR6	No 1394 adapter found error	
ERR7	Flash Update error	
ERR8	Bus error	
ERR9	1394 operation timeout	
ERR10	VME operation timeout	
ERR11	Reserved	
ERR12	Reserved	
ERR13	Reserved	
ERR14	Reserved	
ERR15	Reserved	
ERR16	Reserved	